



UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE

United States Patent and Trademark Office

Address: COMMISSIONER FOR PATENTS

P.O. Box 1450

Alexandria, Virginia 22313-1450

www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/600,284	06/20/2003	Steve Burns	50277-2139	9791
43425 7590 05/25/2010 HICKMAN PALERMO TRUONG & BECKER/ORACLE 2055 GATEWAY PLACE SUITE 550 SAN JOSE, CA 95110-1083				
EXAMINER				
TSUL, WILSON W				
ART UNIT		PAPER NUMBER		
2178				
MAIL DATE		DELIVERY MODE		
05/25/2010		PAPER		

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary

Application No.

10/600,284

Applicant(s)

BURNS ET AL.

Examiner

WILSON TSUI

Art Unit

2178

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --
Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

- 1) ☒ Responsive to communication(s) filed on 08 February 2010.
- 2a) ☒ This action is **FINAL**. 2b) ☐ This action is non-final.
- 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

- 4) ☒ Claim(s) 1-23, 47 and 49-72 is/are pending in the application.
- 4a) Of the above claim(s) _____ is/are withdrawn from consideration.
- 5) ☐ Claim(s) _____ is/are allowed.
- 6) ☒ Claim(s) 1-23, 47 and 49-72 is/are rejected.
- 7) ☐ Claim(s) _____ is/are objected to.
- 8) ☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

- 9) ☐ The specification is objected to by the Examiner.
- 10) ☐ The drawing(s) filed on _____ is/are: a) ☐ accepted or b) ☐ objected to by the Examiner.
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).
Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
- 11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

- 12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
- a) ☐ All b) ☐ Some * c) ☐ None of:
- ☐ Certified copies of the priority documents have been received.
 - ☐ Certified copies of the priority documents have been received in Application No. _____.
 - ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

- 1) ☒ Notice of References Cited (PTO-892)
- 2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
- 3) ☐ Information Disclosure Statement(s) (PTO/SB-06)
Paper No(s)/Mail Date _____
- 4) ☐ Interview Summary (PTO-413)
Paper No(s)/Mail Date _____
- 5) ☐ Notice of Informal Patent Application
- 6) ☐ Other: _____

DETAILED ACTION

1. This final action is in response to the amendment filed on: 02/08/10.
2. Claims 1-5, 18, 49-53, 55, 56, 64, and 66 are amended. Claims 24-46, and 48 are cancelled. Claims 1, 18, 49, and 66 are independent claims. Claims 1-23, 47, and 49-72 are pending.
3. The following rejections are withdrawn, in view of new grounds of rejection necessitated by applicant's amendment:
 - Claims 1-3, 5-14, 16-26, 28- 37, and 39-51, 53-62, and 64-72 rejected under 35 U.S.C. 103(a) as being unpatentable over Abrams et al, in further view of Polizzi et al, and further in view of Holzner.
 - Claims 4, 27, and 52 rejected under 35 U.S.C. 103(a) as being unpatentable over Abrams et al, in view of Polizzi et al, in view of Holzner, and further in view of Hind et al.
 - Claims 15, 38, and 63 rejected under 35 U.S.C. 103(a) as being unpatentable over Abrams et al, in view of in view of Polizzi et al, in view of Holzner, and further in view of Katariya et al.

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

4. Claims 1-3, 5, 7, 8, 13, 14, 16-22, 49-51, 53-62, and 64-72 are rejected under 35 U.S.C. 103(a) as being unpatentable over Abrams et al (US Patent: 6,675,350 B1, issued: Jan. 6, 2004, filed: Nov. 4, 1999), in further view of Schaeck (US Patent: 7,502,833 B2, issued: Mar. 10, 2009, filed: Mar. 12, 2002).

With regards to claim 1, Abrams et al teaches:

- *receiving a request to display the page* (whereas, a user requests a page: Fig 6)
In response to receiving a request to display the page, performing the steps of:
- *Determining that the page is associated with a page parameter* (Fig 6: whereas a page is displayed according to user preference data/parameters for collecting data from a particular site/URL (column 4, lines 13-30))
- *Inspecting a mapping to determine that the page parameter is mapped to a portlet parameter of a portlet* (As shown in Fig 2a, and Fig 2b, and explained in column 4, lines 13-30, the page parameters are checked and refined by a user (thus establishing mapping data with respect to a page parameter and a portlet parameter) to determine what summary information to display in each of the portlets (Fig 6, 630).
- *Passing a value associated with the page parameter as a value of the portlet parameter to a routine responsible for rendering the component, and the routine generating the component based upon the value associated with the portlet parameter* (Fig 2a, and Fig 2b: whereas, the portlets use page parameters such

as URL data to display page/summary information for the page located at the particular URL, and constraint based parameters to display constraint based page/summary information for the page located at the particular URL) *and inserting the component that was generated by the routine into the page* (Fig 6: whereas, the generated data/component is inserted into a page).

- *wherein the steps of the method are performed by one or more computing devices* (column 2, lines 52-65: whereas, the method can be implemented on a display computing device).

However, Abrams et al does not expressly teach

generating and storing a mapping that maps one or more page parameters to one or more portlet parameters, wherein the mapping is stored separate from pages associated with the one or more page parameters; ... determining that the page is associated with a page parameter from the one or more page parameters, using the mapping to determine which portlet parameters are mapped to the page parameter; wherein using the mapping includes retrieving ... the mapping; and wherein the portlet is executable code that is operable to generate page component; passing a value associated with the page parameter to the portlet as a value of the portlet parameter, the portlet generating a component based upon the value that is passed to the portlet as the value of the portlet parameter; and inserting the component that was generated by the portlet into the page;

Yet, Schaeck teaches

generating and storing a mapping that maps one or more page parameters to one or more portlet parameters, wherein the mapping is stored separate from pages associated with the one or more page parameters (whereas, based on a page display request, one or more portlets are identified/requested to generate components of a page (column 4, lines 17-23, lines 45-47). As known in the art, each page can have parameters (column 1, lines 59-61: as known in the art, a page can have associated parameters, as explained in an example of a position and grade of an Employee) and those parameters are sent to a corresponding portlet that generates a component/fragment of a page (column 1, lines 64-65: whereas, as known in the art, portlets can generate fragments/components of a page)); ... determining that the page is associated with a page parameter *from the one or more page parameters* (column 1, lines 59-61: whereas, as known in the art one or more parameters can be associated with a page (i.e. such as the position and grade of an Employee) and the parameters are mapped/bound to a portlet through the use of binding templates (column 5, lines 45-50)), *using the mapping to determine which portlet parameters are mapped to the page parameter* (column 4, lines 16-26: whereas, the binding data further includes the use of a portlet proxy to properly communicate (SOAP) a plurality of page parameters to portlet parameters native to a desired/target portlet); *wherein using the mapping includes retrieving ... the mapping* (column 4, lines 16-26 and column 4, lines 45-55: whereas, binding data used to make appropriate portlet proxy calls is used/retrieved); and *wherein the portlet is executable code that is operable to generate page component*

(column 1, lines 64-65: whereas as known in the art, a portlet can generate a fragment/component of a page); passing a value associated with the page parameter to the portlet as a value of the portlet parameter (whereas, as similarly explained above certain data for a page is passed to a portlet parameter, through mapping data, such that appropriate portlet proxy and SOAP calls are made to the portlet), the portlet generating a component based upon the value *that is passed to the portlet as the value of the portlet parameter* (column 1, lines 61-64: whereas, as known in the art, values can be passed to a portlet, such that the portlet can generate fragment data); and inserting the component that was generated by the *portlet* into the page (Abstract, , Fig 1, column 2, lines 1-3: whereas, as known in the art, a plurality of fragments/components can be inserted/integrated into a portal page);

It would have been obvious to one of the ordinary skill in the art at the time of the invention to have modified Abrams et al's method for displaying a portal aggregation method, such that the mapping information is generated and stored separate from pages and a portlet is identified/called to generate a page component, as similarly explained by Schaeck. The combination would have allowed Abrams et al to have "simplified procedure for installing, accessing, using remote portlets and easily shared portlets" (Schaeck, column 3, lines 5-11).

With regards to claim 2, which depends on claim 1, Abrams et al and Schaeck similarly teaches *wherein: using the mapping further includes determining that the page parameter is mapped to a second portlet parameter associated with a second*

component of the page; and in response to receiving the request to display the web page, further performing the step of passing the value associated with the page parameter as the value of the second portlet parameter to a second portlet that generates the second component, as similarly explained in the rejection for claim 1, whereas page parameter data can be used for a plurality of portlets, to generate a plurality of components).

With regards to claim 3, which depends on claim 1, Abrams et al and Schaeck similarly teach *wherein: the step of generating and storing the mapping further comprises mapping a plurality of page parameters for the web page to a plurality of portlet parameters associated with the component of web page; the step of inspecting the mapping further comprises the step of determining which page parameters of the plurality of page parameters are mapped to each of the plurality of portlet parameters; the step of passing the value further comprises the step of passing, based on the mapping, values associated with the plurality of page parameters as the values of the plurality of portlet parameters to the portlet that generates; and the step of the portlet generating the component further comprises the step of the portlet generating the component based upon the values associated with the plurality of portlet parameters, as similarly explained in the rejection for claim 1 (whereas multiple parameters are supported using binding/template-binding data to pass values to a portlet for component generation), and is rejected under similar rationale.*

With regards to claim 5, which depends on claim 1, Abrams et al and Schaeck similarly teaches wherein the step of generating and storing the mapping comprises *the steps of mapping the page parameter to the portlet parameter and mapping a second page parameter to a second portlet parameter of the portlet that generates the component of the page*, as similarly explained in the rejection for claim 1, URL data is the first page parameter, and constraint based parameters are used as secondary parameters for the component of the page; and thus, rejected under similar rationale.

With regards to claim 7, which depends on claim 1, Abrams et al and Schaeck similarly teach *wherein the request to display the page includes a URL and the URL includes the value associated with the page parameter, and wherein the step of passing the value associated with the page parameter is performed by passing the value contained in the URL as the value of the portlet parameter* (whereas, as explained in column 4, lines 13-30, and in the rejection for claim 1, URL data is used as parameter information, to be passed as the value of the portlet parameter).

With regards to claim 8, which depends on claim 1, Abrams et teaches *further comprising the steps of: presenting to a user a user interface for customizing the page; in response to the user interacting with the user interface, obtaining a user specified value for the page parameter; and wherein the step of passing the value associated with the page parameter is performed by passing the user specified value as the value of the portlet parameter to the portlet responsible for rendering the component*, (in column 4,

lines 1-12, and column 6, lines 25-32, a user interface is used by a user to specify page parameter values including URL, constraint, and layout/positions/fonts to a routine for rendering/displaying the component).

Additionally, as explained in the rejection for claim 1, the combination of Abrams et al and Schaeck's routine is modified such that a portlet-method is used when *passing the value of the portlet parameter to the portlet that generates the component*.

With regards to claim 13, which depends on claim 1, Abrams et al teaches *further comprising the step of presenting to a page designer a user interface for specifying the mapping between the page parameter and the portlet parameter* ((whereas, as explained in column 4, lines 1-12, and column 6, lines 25-32, a user interface is used by a user to specify page parameter values including URL, constraint, and layout/positions/fonts to a routine for rendering/displaying/mapping the component)).

With regards to claim 14, which depends on claim 1, Abrams et al teaches registering the portlet with a portal repository, wherein the process of registering the portlet causes data associated with the portlet to be stored in the portal repository (Abrams et al, claim 1: whereas, a data source comprises registered profile data associated with the routine). Also the combination of Abrams et al and Schaeck teaches registering the portlet with a portal repository, wherein the process of registering the portlet causes data associated with the portlet to be stored in the portal repository

(Schaeck, column 4, lines 35-55: whereas portlets are registered/published in a portal/central registry)

With regards to claim 16, which depends on claim 1, Abrams et al teaches *further comprising the step of receiving input from a page designer through a user interface to create the mapping between the portlet parameter and the page parameter* (whereas, as explained in column 4, lines 1-12, and column 6, lines 25-32, a user interface is used by a user to specify page parameter values including URL, constraint, and layout/positions/fonts to a routine for rendering/displaying/mapping the component).

With regards to claim 17, which depends on claim 1, Abrams et al teaches the method further comprises the step of retrieving the stored value; and the step of the portlet generating the component further comprises the step of the portlet generating the component based upon the retrieved value (claim 1 of Abrams et al, Fig. 2A: whereas, the stored value(s)/preferences/constraints are stored in data stores, which are used to generate the components 240, 250, and 260). Additionally, as explained in the rejection for claim 1, the combination of Abrams et al and Schaeck teaches Abrams et al's routine is modified such that *a portlet-method is used when passing the value of the portlet parameter to the portlet that generates the component*.

With regards to claim 18. Abrams et al teaches a method comprising:

In response to a user manipulating a component associated with a page, a portlet that

generates the component generating a particular event (column 4, lines 20-21: whereas a user manipulates a web address in component 220 of a portlet, causing the portlet in the page to generate a URL selection event)

logic associated with the page; inspecting a first mapping (whereas, as explained in column 4, lines 1-12, and column 6, lines 25-32, a user uses a component/portlet (by generating an event as explained above) to specify page parameter values including URL, constraint, and layout/positions/fonts to a routine for rendering/displaying/mapping the component), *determining, based on the first mapping and the passed data, an action to perform in response to the particular event* (whereas the action to perform is to display all hyperlinks with their associated text for the selected site in pane 260 (column 4, lines 21-24)); *determining, based on the first mapping that an event output parameter associated with the particular event is mapped to a page parameter; and causing the action to be performed ... wherein causing the action to be performed comprises passing a value of the event output parameter as the value of the page parameter; wherein the steps of the method are performed by one or more computing devices* (column 4, lines 21-29: whereas, the URL data that represents the event is mapped to panes 240, 250, and 260, and a display action with regards to the URL data is performed).

However, Abrams et al does not expressly teach *generating and storing a first mapping that maps one or more events to one or more actions and one or more event output parameters to one or more page parameters, wherein the first mapping is stored separate from web pages associated with the one or more page parameters; wherein*

the web pages include a web page; wherein the portlet is executable code that is operable to generate page components; logic associated with the *web page intercepting data, passed by the portlet, that represents the particular event; retrieving* the first mapping.

Yet, Schaek teaches *generating and storing a first mapping that maps one or more events to one or more actions and one or more event output parameters to one or more page parameters, wherein the first mapping is stored separate from web pages associated with the one or more page parameters*; wherein the web pages include a web page; wherein the portlet is executable code that is operable to generate page components (whereas, based on a page display request, one or more portlets are identified/requested to generate components of a page (column 4, lines 17-23, lines 45-47). As known in the art, each page can have parameters (column 1, lines 59-61: as known in the art, a page can have associated parameters, as explained in an example of a position and grade of an Employee) and those parameters can be input events (column 1, lines 62-63), causing a event output action; the output data being sent to a corresponding portlet that generates a component/fragment of a page (column 1, lines 64-65: whereas, as known in the art, portlets can generate fragments/components of a page)); logic associated with the *web page intercepting data, passed by the portlet, that represents the particular event* (column 1, lines 64-65: whereas, the web page is then modified to include/intercept component/fragment data); *retrieving* the first mapping (column 4, lines 16-26 and column 4, lines 45-55: whereas, binding data used to make appropriate portlet proxy calls is used/retrieved).

It would have been obvious to one of the ordinary skill in the art at the time of the invention to have modified Abrams et al's method for displaying a portal aggregation method, such that the mapping information is generated and stored separate from pages and a portlet is identified/called to generate a page component, as similarly explained by Schaeck. The combination would have allowed Abrams et al to have "simplified procedure for installing, accessing, using remote portlets and easily shared portlets" (Schaeck, column 3, lines 5-11).

With regards to claim 19, which depends on claim 18, Abrams et al teaches *wherein the page is a first page and the page parameter is associated with a second page; and the step of causing the action to be performed further comprises the step of passing the value of the page parameter to logic responsible for rendering a second page*, as similarly explained in claim 1, the URL data (value of the page parameter) is passed to logic/routine(s) responsible for rendering an updated/second page, and is rejected under the similar rationale.

With regards to claim 20, which depends on claim 18, Abrams et al teaches *wherein the step of causing the action to be performed further comprises the step of generating a request that specifies a URL, wherein the value of the page parameter is included in the URL*: (whereas, as explained in column 4, lines 13-30, and in the rejection for claim 1, URL data is used as parameter information, to be passed as the value of the portlet parameter).

With regards to claim 21, which depends on claim 20, Abrams et al teaches: *the step of generating the request further comprises the step of generating a request for executable code; and the step of causing the action to be performed further comprises the step of invoking the executable code*, as similarly explained in the rejection for claim 1, page parameter data is passed to the appropriate portlet parameters, and the passing of the value causes the display/render action to be performed. Since the rendering as shown in Fig 2A as performed/executed, the figure inherently teaches that code must have been executed in order for the appropriate components/portlets to have been updated with the mapped parameter values.

With regards to claim 22, which depends on claim 21, Abrams et al teaches *wherein the executable code is a web service* (column 1, lines 45-60: *whereas, the executable code, provides user's with a service to collect information from disparate sources, to be displayed in a summarized and consistent manner*).

With regards to claim 49, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 1, is rejected under similar rationale.

With regards to claim 50, which depends on claim 49, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 2, is rejected under similar rationale.

With regards to claim 51, which depends on claim 49, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more

processors perform a method similar to the method of claim 3, is rejected under similar rationale.

With regards to claim 53, which depends on claim 49, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 5, is rejected under similar rationale.

With regards to claim 54, which depends on claim 49, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 6, is rejected under similar rationale.

With regards to claim 55, which depends on claim 49, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 7, is rejected under similar rationale.

With regards to claim 56, which depends on claim 49, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 8, is rejected under similar rationale.

With regards to claim 57, which depends on claim 49, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 9, is rejected under similar rationale.

With regards to claim 58, which depends on claim 57, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 10, is rejected under similar rationale.

With regards to claim 59, which depends on claim 57, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 11, is rejected under similar rationale.

With regards to claim 60, which depends on claim 57, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 12, is rejected under similar rationale.

With regards to claim 61, which depends on claim 49, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 13, is rejected under similar rationale.

With regards to claim 62, which depends on claim 49, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 14, is rejected under similar rationale.

With regards to claim 64, which depends on claim 49, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more

processors perform a method similar to the method of claim 16, is rejected under similar rationale.

With regards to claim 65, which depends on claim 49, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 17, is rejected under similar rationale.

With regards to claim 66, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 18, is rejected under similar rationale.

With regards to claim 67, which depends on claim 66, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 19, is rejected under similar rationale.

With regards to claim 68, which depends on claim 66, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 20, is rejected under similar rationale.

With regards to claim 69, which depends on claim 68, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 21, is rejected under similar rationale.

With regards to claim 70, which depends on claim 69, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 22, is rejected under similar rationale.

With regards to claim 71, which depends on claim 66, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 23, is rejected under similar rationale.

With regards to claim 72, which depends on claim 49, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 47, is rejected under similar rationale.

5. Claims 4 and 52 are rejected under 35 U.S.C. 103(a) as being unpatentable over Abrams et al (US Patent: 6,675,350 B1, issued: Jan. 6, 2004, filed: Nov. 4, 1999), in view of Schaeck (US Patent: 7,502,833 B2, issued: Mar. 10, 2009, filed: Mar. 12, 2002) and further in view of Hind et al (US Application: 2004/0205555 A1, published: Oct. 14, 2004, filed: Sep. 18, 2001).

With regards to claim 4, which depends on claim 1, Abrams et al teaches wherein the step of generating and storing the mapping comprises the step of *mapping the page parameter to the portlet parameter associated with the component of the page*, as similarly explained in the rejection for claim 1, and is rejected under similar rationale.

However, Abrams et al does not expressly teach ... *without mapping the page parameter to portlet parameters associated with any other components of the page*. Hind et al teaches ... *without mapping the parameters to portlet parameters associated with any other components of the page* (Fig. 3A, paragraph 0024: whereas, content for some components/portlets are updated, while some are not, and thus components/portals are selectively mapped for receiving parameter data). It would have been obvious to one of the ordinary skill in the art at the time of the invention to have modified Abrams et al and Hofmann et al's method for mapping page parameters, to have only mapped page parameters to a select page component, as taught by Hind et al. The combination of Abrams et al, Schaeck, and Hind et al would have allowed Abrams et al to have "reduced the time a user waits for receiving a portal page [by] spawning individual threads for reach portlet" (Hind et al, paragraph 0009).

With regards to claim 52, which depends on claim 49, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 4, is rejected under similar rationale.

6. Claims 6, 9, 10, 11, 12, 23, and 47 are rejected under 35 U.S.C. 103(a) as being unpatentable over Abrams et al (US Patent: 6,675,350 B1, issued: Jan. 6, 2004, filed: Nov. 4, 1999), in view of Schaeck (US Patent: 7,502,833 B2, issued: Mar. 10, 2009,

filed: Mar. 12, 2002), and further in view of Polizzi et al (US Application: US 2002/0052954 A1, published: May 2, 2002, filed: Apr. 27, 2001)

With regards to claim 6, Abrams et al teaches *establishing the page parameter, and passing the value associated with the page parameter further comprises the step of passing the value as the value of the portlet parameter the portlet that generates the component*, as similarly explained in the rejection for claim 1, and is rejected under similar rationale. However, Abrams et al does not expressly teach the value associated with the page parameter is a *default value*.

Yet, Schaeck et al teaches a *default value* (paragraph 0033: *whereas, default values can be associated with page parameters, when retrieving the appropriate portlet*). It would have been obvious to one of the ordinary skill in the art at the time of the invention to have modified Abrams et al and Schaeck's method for mapping a page parameter to a portlet parameter, such that the page parameter value is a default value, as also taught by Schaeck. The combination of Abrams et al, Schaeck, Polizzi et al would have allowed Abrams et al to have "allowed a user to have been able to customize the content of his personal portal page by adding or removing certain portal objects or by modifying the content of certain portal objects" (Polizzi et al, paragraph 0030).

With regards to claim 9, which depends on claim 1, Abrams et al teaches *determining a selected value based on override preferences* (column 6, lines 12-32: *whereas, override*

settings/preferences are determined), and *passing the selected value as the value of the portlet parameter to the routine responsible for rendering the component* (as similarly explained in the claim 1, and also in column 6, lines 12-32, the selected preference values are used to render a customized view).

However Polizzi et al teaches an override *hierarchy*, and passing the user specified value as the value of the portlet parameter to the portlet that generates the component (paragraph 0033: whereas, a user specified value can override a default value of the portlet parameter to the portlet that generates the component).

It would have been obvious to one of the ordinary skill in the art at the time of the invention to have modified. Abrams et al method for using portlets to assemble a page, such that one or more portlets can accept user specified parameters that override previous parameter values, as taught by Polizzi et al. The combination of Abrams et al, Schaeck, and Polizzi et al would have allowed Abrams et al to have have "implemented a standardized, easy-to-learn method for retrieving data through the use of an enterprise-wide computer system, which is connected to a variety of computer systems" (Polizzi et al, paragraph 0003).

With regards to claim 10, which depends on claim 9, Abrams et al teaches *the plurality of values includes a URL page parameter value* (as similarly explained in the rejection for claim 1) *and a customize page parameter value* (as similarly explained in the rejection for claim 1, whereas the constraint based parameters, are custom page parameter values), as well as *override* preferences (column 6, lines 12-32: whereas,

override settings/preferences are determined). However, Abrams et al does not expressly teach an override *hierarchy that specifies that the URL page is the page parameter value is the selected value*.

Yet, the combination of Abrams et al, Schaeck, and Polizzi et al teaches an override hierarchy, and a *default page parameter value as the selected value*, as similarly explained in the rejection for claim 9, and is rejected under similar rationale.

It would have been obvious to one of the ordinary skill in the art at the time of the invention to have modified Abrams et al's page parameter value, such as a URL; such that the override hierarchy is used to specify the value to be a selected value, as taught by Polizzi et al. The combination of Abrams et al, Schaeck, and Polizzi et al would have allowed Abrams et al to have "allowed a user to customize content of his personal page, by adding or removing certain portal objects, or by modifying the content of certain portal objects" (Polizzi et al, paragraph 0030).

With regards to claim 11, which depends on claim 9, Abrams et al teaches *the plurality of values includes a page parameter value* (as similarly explained in the rejection for claim 1) *and a customize page parameter value* (as similarly explained in the rejection for claim 1, whereas the constraint based parameters, are custom page parameter values), as well as *override preferences* (column 6, lines 12-32: whereas, override settings/preferences are determined). However, Abrams et al does not expressly teach an override *hierarchy that specifies that the customize page parameter value is the page parameter value is the selected value*.

Yet, the combination of Abrams et al, Schaeck, and Polizzi et al teaches an override hierarchy that specifies that the customized page parameter value is the page parameter value is the selected value, as similarly explained in the rejection for claim 9, and is rejected under similar rationale.

With regards to claim 12, which depends on claim 9, Abrams et al teaches *the plurality of values includes a page parameter value* (as similarly explained in the rejection for claim 1), *a portlet specified value* (as similarly explained in the rejection for claim 1), as well as *override preferences* (column 6, lines 12-32: *whereas, override settings/preferences are determined*). However, Abrams et al does not expressly teach, the page parameter value is a *default value*, and an *override hierarchy that specifies that the default page parameter is the selected value*.

Yet, the combination of Abrams et al, Schaeck, and Polizzi et al teaches *an override hierarchy that specifies that the default page parameter is the selected value*, as similarly explained in the rejection for claim 9, and is rejected under similar rationale.

With regards to claim 23, which depends on claim 18, Abrams et al teaches wherein:

The action comprises rendering a second page, wherein the page parameter is associated with the second page, and wherein rendering the second page (as similarly

explained in the rejection for claim 19, and is rejected under similar rationale) comprises the steps of:

Inspecting a mapping to determine that the page parameter is mapped to a portlet, as similarly explained in the rejection for claim 1, and is rejected under similar rationale.

The second portlet generating a component based upon the value associated with the portlet parameter, as similarly explained in the rejection for claim 19, and is rejected under similar rationale.

Inserting the second component that was generated by a portlet into a page, as similarly explained in the rejection for claim 19, and is rejected under similar rationale.

However, Abrams et al does not expressly teach Inspecting a *second mapping* to determine that the page parameter is mapped to a portlet parameter of a *second* portlet that generates a *second* component of the *second* page that is based, at least in part, on the portlet parameter; and Passing the value of the page parameter as the value of the portlet parameter to the *second* portlet.

Yet, the Polizzi et al similarly teaches Inspecting a *second mapping* to determine that the page parameter is mapped to a portlet parameter of a *second* portlet that generates a *second* component of the *second* page that is based, at least in part, on the portlet parameter; and Passing the value of the page parameter as the value of the portlet parameter to the *second* portlet (paragraph 0092 of Polizzi et al: whereas, more than one portlet is retrieved to generate a page comprising a plurality of components);

It would have been obvious to one of the ordinary skill in the art at the time of the invention to have modified Abrams et al and Schaena's Portal aggregation method of using a separately stored mapping to map a parameter to generate a first component, such that the mapping would further include mapping the parameter to generate a second component, as taught by Polizzi et al. The combination would have "allowed a user to customized content of his personal page, by adding or removing certain portal objects, or by modifying the content of certain portal objects" (Polizzi et al, paragraph 0030).

With regards to claim 47, which depends on claim 1, the combination of Abrams et al, Schaeck, and Polizzi et al teaches *wherein the portlet is a first portlet and wherein the mapping maps a single page parameter, of the one or more page parameters, to a first portlet parameter of the first portlet, and to a second portlet parameter of a second portlet*, as similarly explained in the rejection for claim 23, and is rejected under similar rationale.

7. Claims 15, and 63 are rejected under 35 U.S.C. 103(a) as being unpatentable over Abrams et al (US Patent: 6,675,350 B1, issued: Jan. 6, 2004, filed: Nov. 4, 1999), in view of Schaeck (US Patent: 7,502,833 B2, issued: Mar. 10, 2009, filed: Mar. 12, 2002), , and further in view of Katariya et al (US Patent: 6,564,251 B2, issued: May 13, 2003, filed: Dec. 3, 1998).

With regards to claim 15, which depends on claim 14, Abrams et al, and Schaeck teaches *the data associated with the portlet, and communicated with the portal repository*, as similarly explained in the rejection for claim 14, and is rejected under similar rationale. However, Abrams et al, and Schaeck do not expressly teach the data associated with the portlet, is communicated to the portal repository *as an XML document*.

Katariya et al teaches communicating with the portal repository, through the use of an *XML document* (columns 5 and 6, lines 59-67 and 1-9 respectively: whereas, preference/parameter information is communicated to a portal repository via XML format).

It would have been obvious to one of the ordinary skill in the art at the time of the invention to have modified Abrams et al's communication of data to a portal repository, such that the data is passed using a XML document, as similarly taught by Katariya et al. The combination of Abrams et al, Schaeck, and Katariya et al would have allowed Abrams et al to have allowed "the content of each page to have been enhanced by the rendered data from the provider objects, thereby adding dynamic behavior to the predefined page" (Katariya et al, column 2, lines 26-31).

With regards to claim 38, for a computer-readable storage medium storing one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform a method similar to the method recited in claim 15, and is rejected under similar rationale.

With regards to claim 63, which depends on claim 62, for a computer readable volatile or non-volatile medium, storing instructions that when executed by one or more processors perform a method similar to the method of claim 15, is rejected under similar rationale.

Response to Arguments

8. Applicant's arguments with respect to claims 1-23, 47, and 49-72 have been considered but are moot in view of the new ground(s) of rejection (due to applicant clarifying how the mapping is used in context with a portal page and portlets).

9. The applicant argues on page 27 of applicant remarks that "... the user-defined organizational structure described in col. 6, lines 12-25 of ABRAMS is used for generating HTML code for formatting and displaying already-extracted summary information/headlines. Thus, any user provided data values for selecting summary information /headlines are not used to generate HTML code that would display the extracted summary information/headlines. Consequently ABRAMS does not describe a code that need any mapping between the user provided input for selecting summary information/headlines and any user-defined organizational structure that is used to generate HTML code for displaying the extracted summary information/headlines.

10. However, the examiner respectfully disagrees and points out that the claim language does not confine/restrict/require the scope of data being generated. Therefore, the HTML code that is generated for formatting and displaying the summary information/headlines (col 6, lines 12-25 of ABRAMS: the display of summary

information/headlines *based on* user input), still reads upon a mapping. The Examiner respectfully directs applicant to the rejection for claim 1 above, which further refers to Schaena to show how the mapping can be improved, when using web based portlets that generate code.

Conclusion

11. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire **THREE MONTHS** from the mailing date of this action. In the event a first reply is filed within **TWO MONTHS** of the mailing date of this final action and the advisory action is not mailed until after the end of the **THREE-MONTH** shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than **SIX MONTHS** from the date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to **WILSON TSUI** whose telephone number is (571)272-7596. The examiner can normally be reached on **Monday - Friday**.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Stephen Hong can be reached on (571) 272-4124. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Stephen S. Hong/
Supervisory Patent Examiner, Art
Unit 2178

/Wilson Tsui/
Patent Examiner
Art Unit: 2178
May 21, 2010.